IN THE CLAIMS

1-53   (Canceled)

54.   (Currently Amended) A machine implemented method performed by a network

element having a first interface communicatively coupled to a first network and a second

interface communicatively coupled to a second network, the method comprising:

        receiving at a configuration manager of the network element a request from a user via

                a command line interface (CLI) of the network element for configuring the

                network element, the request including a transaction for accessing a

                configuration file stored inone or more records of a database that is used to

                route network traffic between the first network and the second network via the

                first and second interfaces, the first network being different than the second

                networkconfigure operations of the network element, wherein the CLI is one of

                a plurality of CLI interfaces provided by the network element to substantially

                concurrently access the network element;

        in response to the request, the configuration manager spawns a session thread to

                associate the CLI with the configuration manager, wherein the associated

                session thread is used by the CLI to communicate messages with the

                configuration manager;

        the configuration manager receiving from the CLI via the session thread one or more

                CLI commands indicating one or more CLI operations of the transaction;

for each CLI operation, the configuration manager invoking a component manager

associated with a type of each CLI operation to process the CLI operations into

one or more database operations operable by the database being accessed;

in response to the request, recording database operations of the request in a transaction

log separated from the database without accessing the database until a commit

command is received from the user via the CLI of the network element; and

performing in response to a commit command from the CLI indicating that the user

has committed to the database operations, transmitting the recorded database

operations of the request from the transaction log to access a record of the

database associated with the request received from the user in response to a

commit command from the CLI indicating that the user has committed to the

requested configuration, wherein the recorded operations are performed within

the database in response to receiving the recorded database operations from the

transaction log.

    ,

55.   (Previously Presented) The method of claim 54, wherein the transaction log comprises

a persistent memory in which content of the transaction log is maintained after the network

element is powered down or rebooted.

56.   (Previously Presented) The method of claim 54, further comprising prior to recording

the operations of the request in the transaction log, acquiring a lock for locking the record of

the database associated with the request to prevent other users from accessing the record of

the database.

57.   (Previously Presented) The method of claim 56, further comprising:

receiving further modification of configuration from the user prior to the commit

command; and

storing the modification in the transaction log without accessing the data base until the

commit command is received from the user upon which the modification of the

configuration is committed from the transaction log to the locked record of

database.

58.   (Currently Amended) The method of claim ~~56~~57, further comprising:

receiving an abort command from the user via the CLI prior to receiving the commit

command; and

in response to the abort command, removing the operations of the request from the

transaction log and releasing the acquired lock without accessing the database.

59.   (Previously Presented) The method of claim 58, wherein after performing the

removing and releasing in response to the abort command, the record of the database remains

substantially identical with respect to the record prior to receiving the request.

60.   (Previously Presented) The method of claim 56, further comprising indicating within

the transaction log that the request is in a committing state while committing the operations of

the request from the transaction log to the locked record of the database.

61.   (Currently Amended) The method of claim ~~60~~54, further comprising ~~indicating within~~

~~the transaction log that the request is in a non-transaction state if operations of committing the~~

~~operations of the request from the transaction log to the database have completed.~~<u>initializing</u>

<u>the configuration manager during an initialization period of the network element, including</u>

> <u>the configuration manager registering with a process manager, the process manager</u>
>
> > <u>managing processes performed by the component managers,</u>
>
> <u>the configuration manager initializing each of the component managers,</u>
>
> <u>each of the component managers registering with the process manager,</u>
>
> <u>the configuration manager storing identifiers (IDs) of the registered component</u>
>
> > <u>managers in a table, wherein the component managers are subsequently</u>
> >
> > <u>invoked by the configuration manager based on the IDs of the component</u>
> >
> > <u>managers.</u>

62.   (Currently Amended) The method of claim 61, further comprising ~~indicating within~~

~~the transaction log that the request is in a transaction state while recording the operations of~~

~~the request in the transaction log before receiving the commit command from the~~

~~user.~~<u>configuration manager initially configuring the network element during the initialization</u>

<u>period of the network by initiating an initialization transaction, including</u>

> <u>the process manager starting a configuration process by sending a message to the</u>
>
> > <u>configuration manager requesting the initialization transaction,</u>
>
> <u>in response to the message, the configuration manager spawning a session thread for</u>
>
> > <u>the configuration process,</u>
>
> <u>the configuration process requesting a transaction ID for the initialization transaction</u>
>
> > <u>from the configuration manager and obtaining a lock on the database,</u>

the configuration process reading configuration operations from a configuration file

and sending each of the configuration operations to the session thread using the

transaction ID while the database is locked,

after all of the configuration operations have been sent to the session thread, the

configuration process sending a commit command to the session thread for the

initialization transaction, and

the configuration process releasing the lock of the database, wherein the operations

sent to the session thread are performed within the database.

63.   (Currently Amended) The method of claim ~~62~~54, further comprising:

in response to the session thread, the CLI sending a message to the configuration

manager indicating starting of a transaction;

the session thread requesting a transaction ID from the configuration manager;

the configuration manager returning the requested transaction ID to the CLI via the

session thread;

the configuration manager setting a state of the session thread to an in-transaction

state;

the CLI receiving a CLI operation from the user and performing a syntax verification

of the CLI operation;

the CLI sending a message having the CLI operation to the session thread including a

value identifying a component manager corresponding to the CLI operation.

~~detecting whether operations of committing the operations of the request from the~~

~~transaction log to the database have stopped resulted from errors of the~~

~~network element; and~~

in response to the detection, renewing performing the operations of the request from

the transaction log to the database while the record of the database is locked.

64. (Currently Amended) The method of claim 63, wherein the detection of whether

operations of committing the operations of the request has stopped resulted from errors is

performed in response to the network element crashes and recovers from the crash. further

comprising:

> determining whether a CLI operation within the message includes one of a commit
>
> > command and an abort command;
>
> if the CLI operation is not one of the commit and abort commands, the configuration
>
> > manager invoking the component manager identified by the message to process
> >
> > the CLI operation;
>
> the component manager performing semantic verification of the CLI operation and
>
> > rejecting the CLI operation if the CLI operation does not pass the semantic
> >
> > verification;
>
> if the CLI operation passes the semantic verification, the component manager
>
> > processing the CLI operation into one or more database operations to be
> >
> > performed within the database.

65. (Currently Amended) The method of claim 64, wherein the detection is performed by

examining within the transaction log whether the request is in the committing state, and

wherein the renewing is performed only if the request is in the committing state.further

comprising:

> the component manager determining whether a lock contention exists;

if the lock contention exists, the component manager notifying the configuration

manager to block further CLI operations from the CLI;

in response to the notification, the configuration manager notifying the CLI regarding

the lock contention;

the CLI prompting the user to either to wait or abort the CLI operation;

in response to an abort command received from the user, the component manager

aborts the CLI operation.

66.   (Currently Amended) The method of claim ~~62~~65, further comprising:

in response to a wait command received from the user, the component manager

continuing blocking the CLI operation;

the component manager periodically checking whether the lock has been released;

if the lock has not been released, the component manager determining whether a

period of time associated with the wait command has expired;

if the wait command has expired, prompting the user for further instructions including

whether the user wants to wait or abort.

~~detecting whether operations of recording the operations of the request within the~~

~~transaction log have stopped resulted from errors of the network element; and~~

~~in response to the detection, removing the request from the transaction log without~~

~~committing to the database.~~

67.   (Currently Amended) The method of claim 66, ~~wherein the detection of whether~~

~~operations of recording the operations of the request within the transaction log has stopped~~

resulted from errors is performed in response to the network element crashes and recovers from the crash.further comprising:

    if the CLI operation includes an abort command, the transaction corresponding to the

        abort command is aborted and the corresponding database operations are

        removed from the transaction log;

    if the CLI operation includes a commit command, the corresponding session thread

        transitioning from the in-transaction state to an in-commit state;

    marking the corresponding database operations in the transaction log as committed;

    determining whether a network resource process for processing the CLI operation has

        died;

    if the network resource process has not died, committing the corresponding database

        operations to the database via a backend procedure call;

    changing the corresponding session thread from the in-commit state to a not-in-

        transaction state when the transaction is completed.

68.   (Currently Amended) The method of claim 67, wherein the detection is performed by examining within the transaction log whether the request is in the transaction state, and wherein the removing is performed only if the request is in the transaction state.further comprising:

    if the network resource process has died, determining whether a current database

        operation is associated with the network resource process;

    if the current database operation is not associated with the network resource process,

        performing database record request and invoking a backend procedure call;

if the current database operation is not associated with the network resource process, performing database record request without invoking a backend procedure call.

69.   (Previously Presented) The method of claim 56, further comprising:

determining whether the lock being acquired is unavailable;

notifying the user via the CLI that the lock is unavailable; and

prompting the user whether the user desires to wait or cancel the request.

70.   (Previously Presented) The method of claim 59, further comprising:

removing the request from the transaction log in response to receiving a cancel

command from the user in response to the prompting; and

in response to receiving a wait command from the user, repeating acquiring the lock

until the lock has been acquired upon which if the commit command has been

received, the request is committed from the transaction log to the locked record

of the database.

71.   (Currently Amended) A machine-readable medium having executable code to cause a

machine to perform a method of a network element having a first interface communicatively

coupled to a first network and a second interface communicatively coupled to a second

network, the method comprising:

receiving at a configuration manager of the network element a request from a user via

a command line interface (CLI) of the network element for configuring the

network element, the request accessing a configuration file stored one or more

records of in a database that is used to route network traffic between the first

~~network and the second network via the first and second interfaces, the first~~

~~network being different than the second network~~ configure operations of the

network element, wherein the CLI is one of a plurality of CLI interfaces

provided by the network element to substantially concurrently access the

network element;

in response to the request, the configuration manager spawns a session thread to

associate the CLI with the configuration manager, wherein the associated

session thread is used by the CLI to communicate messages with the

configuration manager;

the configuration manager receiving from the CLI via the session thread one or more

CLI commands indicating one or more CLI operations of the transaction;

for each CLI operation, the configuration manager invoking a component manager

associated with a type of each CLI operation to process the CLI operations into

one or more database operations operable by the database being accessed;

~~in response to the request,~~ recording database operations ~~of the request~~ in a transaction

log separated from the database without accessing the database until a commit

command is received from the user via the CLI of the network element; and

in response to a commit command from the CLI indicating that the user has committed

to the database operations, transmitting ~~performing~~ the recorded database

operations ~~of the request~~ from the transaction log to ~~access a record of~~ the

database ~~associated with the request received from the user in response to a~~

~~commit command from the CLI indicating that the user has committed to the~~

~~requested configuration,~~ wherein the recorded operations are performed within

the database in response to receiving the recorded database operations from the

transaction log.

72.   (Previously Presented) The machine-readable medium of claim 71, wherein the

transaction log comprises a persistent memory in which content of the transaction log is

maintained after the network element is powered down or rebooted.

73.   (Previously Presented) The machine-readable medium of claim 71, wherein the

method further comprises prior to recording the operations of the request in the transaction

log, acquiring a lock for locking the record of the database associated with the request to

prevent other users from accessing the record of the database.

74.   (Previously Presented) The machine-readable medium of claim 73, wherein the

method further comprises:

    receiving further modification of configuration from the user prior to the commit

        command; and

    storing the modification in the transaction log without accessing the data base until the

        commit command is received from the user upon which the modification of the

        configuration is committed from the transaction log to the locked record of

        database.

75.   (Currently Amended) The machine-readable medium of claim ~~73~~74, wherein the

method further comprises:

receiving an abort command from the user via the CLI prior to receiving the commit

command; and

in response to the abort command, removing the operations of the request from the

transaction log and releasing the acquired lock without accessing the database.

76.   (Previously Presented) The machine-readable medium of claim 75, wherein after

performing the removing and releasing in response to the abort command, the record of the

database remains substantially identical with respect to the record prior to receiving the

request.

77.   (Previously Presented) The machine-readable medium of claim 73, wherein the

method further comprises indicating within the transaction log that the request is in a

committing state while committing the operations of the request from the transaction log to

the locked record of the database.

78.   (Currently Amended) The machine-readable medium of claim ~~77~~71, wherein the

method further comprises ~~indicating within the transaction log that the request is in a non-~~

~~transaction state if operations of committing the operations of the request from the transaction~~

~~log to the database have completed.~~ initializing the configuration manager during an

initialization period of the network element, including

the configuration manager registering with a process manager, the process manager

managing processes performed by the component managers,

the configuration manager initializing each of the component managers,

each of the component managers registering with the process manager,

the configuration manager storing identifiers (IDs) of the registered component

managers in a table, wherein the component managers are subsequently

invoked by the configuration manager based on the IDs of the component

managers.

79.   (Currently Amended) The machine-readable medium of claim 78, wherein the method

further comprises ~~indicating within the transaction log that the request is in a transaction state~~

~~while recording the operations of the request in the transaction log before receiving the~~

~~commit command from the user.~~ configuration manager initially configuring the network

element during the initialization period of the network by initiating an initialization

transaction, including

the process manager starting a configuration process by sending a message to the

.configuration manager requesting the initialization transaction,

in response to the message, the configuration manager spawning a session thread for

the configuration process,

the configuration process requesting a transaction ID for the initialization transaction

from the configuration manager and obtaining a lock on the database,

the configuration process reading configuration operations from a configuration file

and sending each of the configuration operations to the session thread using the

transaction ID while the database is locked,

after all of the configuration operations have been sent to the session thread, the

configuration process sending a commit command to the session thread for the

initialization transaction, and

the configuration process releasing the lock of the database, wherein the operations

sent to the session thread are performed within the database.

80. (Currently Amended) The machine-readable medium of claim ~~79~~71, wherein the
method further comprises:

in response to the session thread, the CLI sending a message to the configuration

manager indicating starting of a transaction;

the session thread requesting a transaction ID from the configuration manager;

the configuration manager returning the requested transaction ID to the CLI via the

session thread;

the configuration manager setting a state of the session thread to an in-transaction

state;

the CLI receiving a CLI operation from the user and performing a syntax verification

of the CLI operation;

'the CLI sending a message having the CLI operation to the session thread including a

value identifying a component manager corresponding to the CLI operation.

~~detecting whether operations of committing the operations of the request from the~~

~~transaction log to the database have stopped resulted from errors of the~~

~~network element; and~~

~~in response to the detection, renewing performing the operations of the request from~~

~~the transaction log to the database while the record of the database is locked.~~

81. (Currently Amended) The machine-readable medium of claim 80, wherein ~~the~~

~~detection of whether operations of committing the operations of the request has stopped~~

~~resulted from errors is performed in response to the network element crashes and recovers~~ ~~from the crash.~~the method further comprises:

determining whether a CLI operation within the message includes one of a commit

command and an abort command;

if the CLI operation is not one of the commit and abort commands, the configuration

manager invoking the component manager identified by the message to process

the CLI operation;

the component manager performing semantic verification of the CLI operation and

rejecting the CLI operation if the CLI operation does not pass the semantic

verification;

if the CLI operation passes the semantic verification, the component manager

processing the CLI operation into one or more database operations to be

performed within the database.

82.   (Currently Amended) The machine-readable medium of claim 81, wherein ~~the~~

~~detection is performed by examining within the transaction log whether the request is in the~~

~~committing state, and wherein the renewing is performed only if the request is in the~~

~~committing state.~~the method further comprises:

the component manager determining whether a lock contention exists;

if the lock contention exists, the component manager notifying the configuration

manager to block further CLI operations from the CLI;

in response to the notification, the configuration manager notifying the CLI regarding

the lock contention;

the CLI prompting the user to either to wait or abort the CLI operation;

in response to an abort command received from the user, the component manager

aborts the CLI operation.

83.   (Currently Amended) The machine-readable medium of claim ~~79~~82, wherein the

method further comprises:

in response to a wait command received from the user, the component manager

continuing blocking the CLI operation;

the component manager periodically checking whether the lock has been released;

if the lock has not been released, the component manager determining whether a

period of time associated with the wait command has expired;

if the wait command has expired, prompting the user for further instructions including

whether the user wants to wait or abort.

~~detecting whether operations of recording the operations of the request within the~~

~~transaction log have stopped resulted from errors of the network element; and~~

~~in response to the detection, removing the request from the transaction log without~~

~~committing to the database.~~

84.   (Currently Amended) The machine-readable medium of claim 83, wherein ~~the~~

~~detection of whether operations of recording the operations of the request within the~~

~~transaction log has stopped resulted from errors is performed in response to the network~~

~~element crashes and recovers from the crash.~~the method further comprises:

if the CLI operation includes an abort command, the transaction corresponding to the

abort command is aborted and the corresponding database operations are

removed from the transaction log;

if the CLI operation includes a commit command, the corresponding session thread

transitioning from the in-transaction state to an in-commit state;

marking the corresponding database operations in the transaction log as committed;

determining whether a network resource process for processing the CLI operation has

died;

if the network resource process has not died, committing the corresponding database

operations to the database via a backend procedure call;

changing the corresponding session thread from the in-commit state to a not-in-

transaction state when the transaction is completed.

85.   (Currently Amended) The machine-readable medium of claim 84, wherein ~~the~~

~~detection is performed by examining within the transaction log whether the request is in the~~

~~transaction state, and wherein the removing is performed only if the request is in the~~

~~transaction state.~~the method further comprises:

if the network resource process has died, determining whether a current database

operation is associated with the network resource process;

if the current database operation is not associated with the network resource process,

performing database record request and invoking a backend procedure call;

if the current database operation is not associated with the network resource process,

performing database record request without invoking a backend procedure call.

86.   (Previously Presented) The machine-readable medium of claim 83, wherein the

method further comprises:

determining whether the lock being acquired is unavailable;

notifying the user via the CLI that the lock is unavailable; and

prompting the user whether the user desires to wait or cancel the request.

87.   (Previously Presented) The machine-readable medium of claim 86, wherein the method further comprises:

removing the request from the transaction log in response to receiving a cancel

command from the user in response to the prompting; and

in response to receiving a wait command from the user, repeating acquiring the lock

until the lock has been acquired upon which if the commit command has been

received, the request is committed from the transaction log to the locked record

of the database.

88.   (Canceled)